C/S架构

客户端请求服务器端， 服务器端接收数据并响应

客户端发送不同的请求到服务器端， 根据不同的请求数据给出不同的响应内容

如果请求越来越多越来越复杂， 服务器端会越来越多，越来越复杂

1、接收客户端请求，获取请求数据（固定的套路）

2、根据不同的请求找相应的资源（找资源的过程是固定的， 资源不是固定的）

3、响应数据给客户端 （固定的）

由此产生了真正的服务器（专门用来做固定套路的步骤）

1、接收客户端的请求

2、查找资源（分发请求）（有不同的资源来处理不同的请求， 我们写的处理）

3、将找到的（处理好的）资源相应（回传）给客户端

服务器会对请求和相应做一定的封装

请求： 连同请求数据和请求信息（客户端信息） Request的对象当中

处理请求的过程以及给出响应（重点）：

规定：能够处理请求的类必须实现（规范）Servlet接口，该接口中已经定义好了一个方法，service方法 （Request, Response）

响应： 将响应的内容封装到一个对象 Response 中

客户端请求服务器 –》 服务器封装 + 小程序（程序员书写、处理） + 资源整合 –》 服务器响应回给客户端。

需要有服务器端(server) ：等待被请求，需要暴露 ip 和 port

需要有客户端(client)：主动发起请求 ，知晓服务端的 ip 和 port

通信规则（协议）：TCP/IP 协议

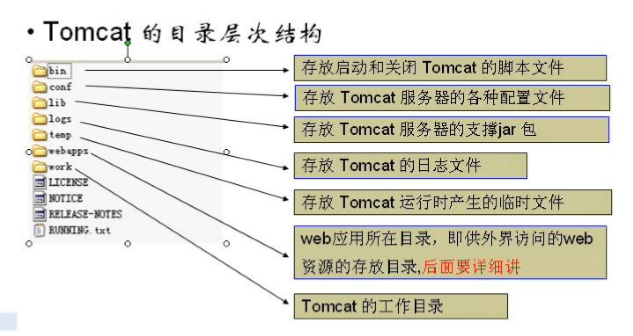
统一服务器端（总开关|总入口） -> Tomcat

统一客户端（所有请求不需要再单独写一套客户端） -> 浏览器

协议名：//域名(ip地址):端口/url 例如：http://www.baidu.com

端口号：浏览器默认端口 80 tomcat：默认端口 8080

Tomcat 是一个符合 JavaEE WEB 标准的最小的 WEB 容器，所有的 JSP 程序一定要有 WEB 容器的支持才能运行，而且在给定的 WEB 容器里面都会支持事务处理操作。



Servlet 类的实现

继承 HttpServelt类 （实现Servlet接口, 继承GenericServlet类）

重写service方法（doGet， doPost）

service方法中有两个参数

Request : 请求对象🡪 请求参数（请求数据）

Response: 响应对象 🡪 响应数据

Xml配置

<Servlet>

<servlet>

<servlet-name>servlet01</servlet-name>

<servlet-class>helloweb.HelloWebServlet</servlet-class>

</servlet>

<servlet-mapping>

<servlet-name>servlet01</servlet-name>

<url-pattern>/s01</url-pattern>

</servlet-mapping>

Servlet的生命周期：借助生命周期方法来推断servlet的生命， 随着servelt的生命迹象来由服务器自动调用的

init -> 初始化

service -> 服务

destroy -> 消亡

出生（初始化）：默认情况第一次访问某个servlet，该servlet会被初始化（一次）单例

活动，提供服务：每次请求某个servlet的时候，都会有活动迹象（会执行多次）

死亡（销毁）：服务器关闭时，servlet会被销毁（一次）

servlet的配置， html文件的放置， 服务器中规划 🡪 能够正确存放的资源，怎么存以及怎么在地址栏中输入路径访问

服务器怎么处理（接收）请求

1、找到对应的文件（html,servlet）

2、如果是servlet还需要确定方法的调用

service , doGet, doPost

发送请求：

1、地址栏直接输入 get

2、form表单 method 🡺 get post

3、超链接 get

区分请求方式并且调用对应的方法：

如果有service存在， 那么就执行service

如果没有service存在， 那么必须存在对应的get或者post方法

Get 数据量小 不安全 直接拼接在浏览器地址栏

Post 数据量大 安全